11-Static Variable

What is a Static Variable?

The static keyword in Java is primarily used for memory management. It can be applied to variables, methods, blocks, and nested classes. The static keyword belongs to the class itself rather than to instances of the class, meaning it's shared across all instances.

The static keyword can be used with:

1. **Variable** (also known as a class variable)
2. **Method** (also known as a class method)
3. **Block**
4. **Nested class**

Java Static Variable

When you declare a variable as static, it is known as a static variable. A static variable is used to refer to the common property of all objects of a class (which is not unique for each object), such as the company name of employees, or the college name of students.

- The static variable gets memory only once in the class area, at the time of class loading.

Advantages of Static Variables:

- **Memory Efficiency:** It saves memory because static variables are allocated memory only once, and the memory is shared across all instances of the class.

Example: Understanding Static Variables

```
class Mob {
    String brand;
    int price;
    String name;

    public void show() {
        System.out.println(brand + "," + price + "," + name);
    }
}
class Demo {
    public static void main(String[] args) {
        Mob obj = new Mob();
```

```
        Mob obj1 = new Mob();
        obj.brand = "Apple";
        obj.price = 1500;
        obj.name = "SmartPhone";

        obj1.brand = "Samsung";
        obj1.price = 1700;
        obj1.name = "SmartPhone";

        obj.show();
        obj1.show();
    }
}
```

Output:

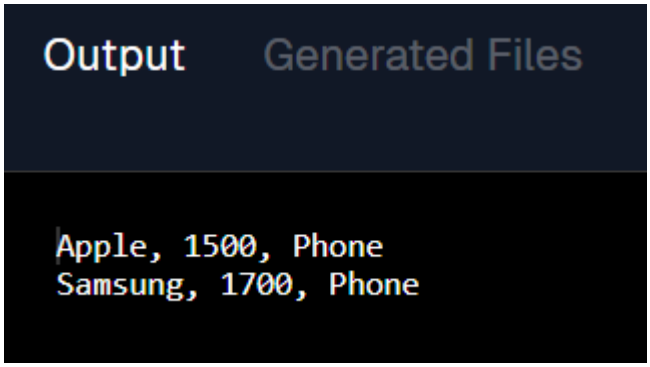
```
Apple, 1500, SmartPhone
Samsung, 1700, SmartPhone
```

In the example above, the name of the device is common for both objects. However, space is consumed each time the name is stored. Wouldn't it be better if we had a way to declare it once and have it remain common for all instances? Yes, we can achieve this by using the static keyword.

Refactoring with Static Variables

By declaring the name variable as static, it becomes a class variable, meaning memory will be allocated only once for it. When the class is loaded, memory is assigned to all static members of the class. If any variable is declared as static, its value will be shared by all objects of the class.

```java
class Mob {
    String brand;
    int price;
    String network;
    static String name;

    public void show() {
        System.out.println(brand + ", " + price + ", " + name);
    }
}

public class Demo {
    public static void main(String[] args) {
        Mob obj = new Mob();
        Mob obj1 = new Mob();

        obj.brand = "Apple";
        obj.price = 1500;
        Mob.name = "SmartPhone";

        obj1.brand = "Samsung";
        obj1.price = 1700;
        Mob.name = "SmartPhone";

        obj.name="Phone";

        obj.show();
        obj1.show();
    }
}
```



Explanation:

- The static keyword ensures that the name variable is shared across all instances of the class. Therefore, when the name is changed for one object, it reflects for all objects.
- The correct way to access static members is by using the class name, as shown with Mob.name.